Expert Systems With Applications 212 (2023) 118716

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Predicting Vehicle Behavior Using Multi-task Ensemble Learning

Reza Khoshkangini [a,*], Peyman Mashhadi [b], Daniel Tegnered [c], Jens Lundström [b], Thorsteinn Rögnvaldsson [b]

[a] Department of Computer Science and Media Technology, Malmö University, Sweden
[b] Center for Applied Intelligent Systems Research (CAISR), Halmstad University, Sweden
[c] Volvo Group Connected Solutions, Gothenburg, Sweden

ARTICLE INFO

Keywords:
Behavior modeling
Multi-task learning
Deep neural networks
Ensemble learning

ABSTRACT

Vehicle utilization analysis is an essential tool for manufacturers to understand customer needs, improve equipment uptime, and to collect information for future vehicle and service development. Typically today, this behavioral modeling is done on high-resolution time-resolved data with features such as GPS position and fuel consumption. However, high-resolution data is costly to transfer and sensitive from a privacy perspective. Therefore, such data is typically only collected when the customer pays for extra services relying on that data. This motivated us to develop a multi-task ensemble approach to transfer knowledge from the high-resolution data and enable vehicle behavior prediction from low-resolution but high dimensional data that is aggregated over time in the vehicles.

This study proposes a multi-task snapshot-stacked ensemble (MTSSE) deep neural network for vehicle behavior prediction by considering vehicles' low-resolution operational life records. The multi-task ensemble approach utilizes the measurements to map the low-frequency vehicle usage to the vehicle behaviors defined from the high-resolution time-resolved data. Two data sources are integrated and used: high-resolution data called Dynafleet, and low-resolution so-called Logged Vehicle Data (LVD). The experimental results demonstrate the proposed approach's effectiveness in predicting the vehicle behavior from low frequency data. With the suggested multi-task snapshot-stacked ensemble deep network, it is shown how low-resolution sensor data can highly contribute to predicting multiple vehicle behaviors simultaneously while using only one single training process.

1. Introduction

High-resolution time-resolved data such as GPS position is costly to transfer from connected assets and is sensitive from a privacy perspective. If similar insights regarding vehicle behavior can be made from lower frequency aggregate data, the size of the population can increase for a similar transfer cost, and the risk of privacy breaches is diminished. This study investigates if different vehicle behaviors determined from high-resolution data in Volvo's fleet management system Dynafleet can also be inferred from low-resolution aggregate Logged Vehicle Data.

Utilization analysis is essential to understand customers' needs and advise future truck developments. For example, the night stop behavior, whether the vehicle usually stops at a single home base, several distant bases, or is irregular, can be used to inform the design of future electric trucks and their infrastructure. To determine this behavior, time-resolved position data needs to be collected for a long time to produce valuable statistics. Due to the cost, only a fraction of the Volvo Trucks customers opt for the more advanced fleet management systems that involve transferring time-resolved data like GPS position, odometer, and fuel consumption through telematics. This means that valuable data is missing when analyzing customer data, leading to possible biases. Coverage can vary significantly depending on the market and truck variants. If the customers who order more expensive trucks are more likely to also have the fleet management system, there can be erroneous conclusions that most trucks are utilized similarly to the more expensive ones. However, many technical parameters are also stored on the vehicles in an aggregate fashion. Either as standalone values, like the total life of vehicle mileage or as 1D or 2D histograms such as the time spent at certain engine speeds and engine torques. There are hundreds of such parameters, either transferred through telematics at different intervals, ranging from days to weeks, or downloaded when the vehicle visits a workshop. Since all vehicles that

* Corresponding author.
E-mail addresses: reza.khoshkangini@mau.se (R. Khoshkangini), peyman.mashhadi@hh.se (P. Mashhadi), daniel.tegnered@volvo.com (D. Tegnered), jens.r.lundstrom@hh.se (J. Lundström), thorsteinn.rognvaldsson@hh.se (T. Rögnvaldsson).

https://doi.org/10.1016/j.eswa.2022.118716
Received 14 April 2022; Received in revised form 28 July 2022; Accepted 26 August 2022
Available online 15 September 2022

upload the detailed time-resolved data also upload the low-resolution aggregate data, a combined data set can be constructed including the behavioral labels that can presently only be determined from the time-resolved data. If these labels can be transferred to the rest of the population, biases can be removed in the utilization analysis. Furthermore, if utilization analysis through this transfer approach can be run only on the aggregate data, many privacy issues can be avoided since the potentially sensitive position data is not needed.

In the automotive sector, lots of studies have been done to increase the safety, sustainability, and performance of the vehicle by modeling driver behaviors, such as drivers' fatigue (Chen, Wang, He, Wang, & Wang, 2022), lane changing (Miyajima & Takeda, 2016), and aggressiveness (Shahverdy, Fathy, Berangi, & Sabokrou, 2020) and mapping them to energy consumption, unhealthy and breakdowns (Alizadeh, Rahimi, & Ma, 2022; Lattanzi & Freschi, 2021), and CO2 emission (Mondal & Gupta, 2022; Powell, Cezar, & Rajagopal, 2022; Xu, Zheng, & Yang, 2021). Lattanzi and Freschi (2021) explore two machine learning techniques – Support Vector Machines (SVM) and Feed-forward Neural Network – on vehicles' descriptive sensor data such as vehicle speed, engine speed, and engine load to identify safe and unsafe driving behaviors. The correlations between driving behaviors and NOx emissions, considering selective catalytic reduction (SCR) and ammonia creation and conversion technology (ACCT) systems, have been studied in Gao et al. (2021). In Prakash and Bodisco (2019), researchers analyzed different levels of aggressive driving and measured NOx emission when the vehicles were driven on a hilly and calm route. Similar considerations were done in Choi and Kim (2017) to expose the effect of aggressive acceleration on fuel consumption. We have observed studies concerning environmental factors such as weather, time, region, and road conditions, to model vehicle behavior and performance (Bousonville, Dirichs, & Krüger, 2019). Although the works described vary widely in terms of what aspects of vehicle/driver behavior that are considered, what problems to solve, and techniques developed, in general machine learning has proved to be successful to model vehicle behavior in the automotive industry (Alizadeh et al., 2022; Lattanzi & Freschi, 2021). However, there is still room to improve the existing modeling systems regarding prediction accuracy, scale, explainability, and long-term utilization and behavior. We believe that such modeling operations can be built and improved by incorporating AI and advanced deep neural network approaches.

Motivated by the above, in this study we develop an ensemble learning method to build models for vehicle behavior prediction. The proposed approach consists of two main modules; the first module is *Data pre-processing*, which involves extracting hidden information from low-frequency truck internal signals. This information is thereafter integrated with vehicle behaviors (computed from high-resolution location data) which also serves as prediction variables—one for each prediction task; In the second module, which includes two phases, an ensemble deep multi-task neural network is used to map the low-resolution but high dimensional data (histograms and tabular data–predictors, measurements, and features refer to the same concept–sensors data) to the vehicle behavior. This module has been designed to carry a *Multi-task Snapshot-stacked Ensemble* approach to increase the expressive capability of learning predictive models. In the first phase, several training learners are built from different perspectives to solve the complex vehicle behavior prediction using vehicle sensor data. This is done by horizontally aggregating the output of several snapshots models built by utilizing a Cyclic Cosine Annealing Schedule (CCAS) (Fu, Li, Liu, Gao, Celikyilmaz et al., 2019) concerning the principle of diversity to improve the predictive model performance. This results – in the second phase – in a meta-multi-task learning capable of learning from various models to increase the accuracy of the performance of behavior modeling. In each phase, a multi-task neural deep network is constructed to train multiple tasks simultaneously by exploiting shared representation in the hidden layers to learn similar knowledge within a set of each task (Zhang & Yang, 2021). In this learning process, the knowledge in each task can be taken and transferred by other related and even unrelated tasks to improve the generalization performance. This encouraged us to study extracting the transferred knowledge between the tasks and quantifying the effect of each task on each other over the training process. Furthermore, at the same time, the shared representation will yield faster training speeds considering our real large-scale data.

This work is the first one to investigate the modeling and prediction of vehicle behaviors by transferring knowledge from high to low-resolution data using multi-task snapshot-stacked ensemble learning. Earlier vehicle behavior estimations have been rule-based, taking into consideration only high-resolution data; however, applying ensemble deep neural networks through multi-task learning on low-resolution sensor data automates this process through one single training process with less effort and more accuracy that can be generalize to various time and vehicle production. The following research questions (RQs) further elaborate the investigative objectives of our proposed approach:

- **RQ1—Vehicle Behavior Modeling:** To what extent could the behavior of vehicles be modeled and predicted from the low-resolution data using the multi-task ensemble deep neural network?
- **RQ2—Vehicle Behavior Transference:** To what extent could the vehicle behavior associated with one task affect the prediction performance of the other tasks over the training process?
- **RQ3—Vehicle Behavior Grouping:** Will vehicle behavior grouping enhance the performance of the predictive multi-task model?

Taken together, these research objectives, our study seeks to counter the practice mentioned above by mapping the vehicle's low-resolution operational usage to long-term behaviors wherein; First, we concentrate on modeling and predicting multiple behaviors simultaneously via a multi-task learning fashion. Since ensemble solutions demonstrated their capability to tackle complex modeling challenges in various domains, we hypothesize that the ensemble deep neural networks approach will be able to overcome this issue. Thus, to answer RQ1 we develop a multi-task ensemble deep neural network by adapting and coupling it with a stacking approach through two phases of generating multiple diverse multi-task deep networks and employing them to build multi-task meta-learning. The multi-task meta-learning is then utilized for the final vehicle behavior prediction. The reported figures demonstrate how this development enhanced the prediction performance compared with other multi-task and ensemble-based ML approaches. Second, we focus on the knowledge transference between the tasks. Indeed, the primary motivation for RQ2 is to quantify the knowledge – positively or negatively – transferred between different tasks over the training process. Given the RQ2 results, RQ3 provides the ability to understand which tasks should be trained together to improve the predictive model's performance. The development of this multi-task ensemble approach in automotive sector – particularly – vehicle behavior modeling our research outlines and constitutes this paper's main contribution. Below are the contributions in this study:

- Concerning previous studies, which mainly use limited features to model vehicle behavior and map to fuel consumption using high-resolution data, we transfer knowledge from high-frequency data to low-resolution data and utilize hundreds of sensors data to model and predict multiple vehicle behaviors at the same time.
- Considering the low classification accuracy of vehicle behavior modeling exploiting vehicle usage, a snapshot-stacked ensemble multi-task deep learning approach is proposed in this study. Compared with the existing algorithms, our method can effectively improve the modeling performance.
- Concerning the ensemble approach, our solution is novel since it adapts the stacking and cosine annealing process to the ensemble structure to build multi-task meta-learning and map the usage to the vehicles' behaviors. In this course, our ensemble benefits from the knowledge transference between tasks to increase predictive performance.

- We adapt and develop an inter-task transference measurement associated with our multi-task ensemble approach to extract and reveal the effect and transference of the tasks on each other over the training process.
- Regarding the application domain, this is the first study in the automotive industry to investigate modeling vehicle behaviors using multi-task ensemble learning on a large scale with thousands of vehicles.

The remainder of the paper is organized as follows; Section 2 presents the related studies in this context. In Section 3, we describe the data used. Section 4 formulates the problem, The proposed multi-task ensemble approach is described in Section 5. Section 6 covers the experimental evaluation and results. A discussion and summary of the work are given in Section 7.

2. Related work

The aim is to detect vehicle long term behavior by using multi-task and ensemble learning. The proposed approach is here placed in context with earlier work on vehicle behavior modeling, multi-task learning, and ensemble methods.

2.1. Vehicle behavior modeling

During the last decades, driving behavior modeling turned into a hot topic in automotive sectors aiming to address various challenges. Earlier studies focus on designing predictive models to create a self-driving system (Badue et al., 2021; Pentland & Liu, 1999), and later studies focus more on modeling drivers' maneuver patterns and lane changing (Lin, Zong, Tomizuka, Song, Zhang et al., 2014; Liu, Kurt, & Özgüner, 2014; Yao, Zhao, Davoine, & Zha, 2012). Recently, more attention has been paid to modeling vehicle/driver behaviors such as drivers' fatigue (Chen et al., 2022), lane changing (Miyajima & Takeda, 2016), aggressiveness (Shahverdy et al., 2020) aiming to support and increase the vehicle performance and sustainability such as energy consumption, breakdowns, CO2 emission (Mondal & Gupta, 2022; Powell et al., 2022; Xu et al., 2021). For example, Gao et al. (2021) explored the correlations between driving behaviors and NOx emissions based on selective catalytic reduction (SCR) and ammonia creation and conversion technology (ACCT) systems. Similarly, Prakash and Bodisco (2019) investigated NOx emissions from a diesel vehicle in a hilly and calm route at different levels of aggressive driving. Regardless of whether the testing route was predominantly flat or extremely hilly, all the driving dynamics parameters and the cumulative NOx emissions were positively correlated with one another. The results suggest that driver aggressiveness is also responsible for the higher levels of NOx emissions. Choi and Kim (2017) studied the impact of acceleration on the increments in fuel consumption and determined the critical aggressive acceleration for starting and driving vehicles.

So far, most approaches have focused on signal processing on individual vehicle sensor data to determine instantaneous behaviors such as driver frustration or lane changing (Miyajima & Takeda, 2016) for developing ADAS systems (Marina Martinez, Heucke, Wang, Gao, & Cao, 2018; Wang et al., 2020) or predicting fuel consumption (Xu, Wei, Easa, Zhao, & Qu, 2018). This is distinct from our unique investigation, which introduces vehicles' long-term behavior modeling approach based on transferring knowledge from high-resolution to low-resolution data.

2.2. Multi-task learning

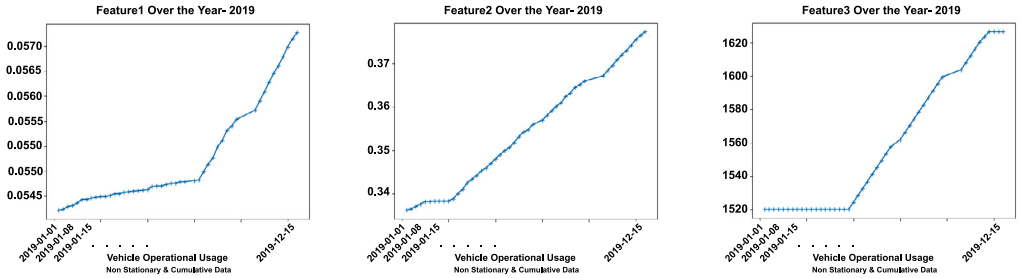
Multi-task learning is a technique utilized to leverage the knowledge shared between multiple related tasks. One of the central promises of multi-task learning is that the shared representation has more generalization power since it needs to learn a more general representation useful for multiple tasks. The use of multi-task learning has been on the rise in the vehicular domains in the past few years, and it has been used in different contexts, including predictive maintenance, autonomous driving, drivers' behavior reasoning, to name just a few (Chowdhuri, Pankaj, & Zipser, 2019; Xie, Hu, Li, & Guo, 2021; Xing, Lv, Cao, & Velenis, 2020; Xun, Liu, & Shi, 2020). For example, Chowdhuri et al. (2019) proposed a unified multi-scale, multi-task learning for drivers' behavior recognition. Different body postures and mental behaviors were adapted to a deep learning-based multi-task framework. A CNN-based model formed the encoder of the model, followed by multiple decoders consisting of Fully connected and LSTM layers branched to different tasks.

An interesting work has been done by Xun et al. (2020) by adopting multi-task learning for three purposes: illegal driver detection, legal driver identification, and driving behavior evaluation. They collected driving data from on-board diagnostic systems installed on different vehicles. The common representation between the three different tasks is generated through a LSTM network which is further followed by a SVDD model for illegal driver detection and two feed-forward neural networks for legal driver identification and driver behavior evaluation. It is worth mentioning that the output of the SVDD model is also used as the input of the driver identification model for the legal cases. More recently, Li, Gong, Lu, and Yi (2021) devised a graph-neural-network-based multi-task learning to predict trajectories of traffic actors with interactive behaviors. Trajectory predictions, 3-D bounding box prediction, and interactive events recognition are considered as multiple tasks. They embed multi-task learning in a Graph-Neural-Network architecture equipped with a multi-task loss function.

2.3. Ensemble approach

Among the vast numbers of approaches introduced and developed for various prediction tasks in many applications, ensemble techniques are quite successful. Ensemble machine learning approaches are techniques that integrate several base ML models to provide improved performance (Dong, Yu, Cao, Shi, & Ma, 2020; Li, Wu, Hu, & Terpenny, 2019). These techniques can be divided into three categories: Stacking, Bagging, and Boosting. The Stacking or Stacked generalization technique (Zhou, 2021) uses and combines prediction from multiple predictive models to build a new model on the training data. Then the model is utilized for the final prediction of the test data. Bagging is a technique seeking a mixed group of ensemble predictive models built by various parts of the training data (González, García, Del Ser, Rokach, & Herrera, 2020). Then statistical approaches such as averaging or voting are used to combine the prediction results from the members. Boosting is another ensemble approach that creates a strong predictive model from several weak classifiers (Mosavi, Sajedi Hosseini, Choubin, Goodarzi, Dineva et al., 2021). In this approach, the training data is used to build a predictive model. The second model then tries to correct the mistakes from the first classifier. This process will continue until the maximum number of predictive models or reach the perfect prediction.

In the context of vehicle behavior modeling, there are a few interesting studies which utilize ensemble approaches. For example, an ensemble semi-supervised solution is introduced in Zhang and Fu (2021) to model the maneuvering behaviors of the vehicles based on data provided by the Next Generation Simulation (NGSIM) project. Using a similar dataset, an ensemble of Boosting and Bagging approach was developed in Hou, Edara, and Sun (2015) for modeling lane changes in order to build a lane change assistance system. For the similar modeling, in Xing, Lv, Wang, Cao, and Velenis (2020) a combination of recurrent neural networks (RNN) and Long Short-Term Memory (LSTM) is proposed based on four critical moments of the lane change process throughout driving on a highway. Researchers working on applications from different sectors also took advantage of ensemble methods. In the context of claim and fault detection, there are some interesting studies in different sectors (Le, Yao, Miller, & Tsao, 2020; Yang, Shao, & Bian,



(a) feature1-over year 2019 (b) feature2-over year 2019 (c) feature3-over year 2019

Fig. 1. Illustration of the non-stationary vehicle operational data, which are collected as scalar features. These three cumulative features (out of hundreds of parameters) are logged over 2019 and indicate that the usage is monotonously increased over the year.

2022). For instance, a bagging ensemble-based approach is developed by Mousavi, Moradi, Chaibakhsh, Kordestani, and Saif (2020) to forecast faults in a single-shaft industrial gas turbine. The fault prognosis was done based on weighted voting aiming to improve the robustness of the fault detection technique. To tackle the same problem in a similar domain, Zhang and Gao (2021) use a deep neural network ensemble approach to a wind turbine data series. In this work, the network is trained offline based on a training set of the imagination matrix that is transformed from data series through segmentation technology. Then the network is retrained by transfer learning based on the renewed training set to achieve enough knowledge to forecast the fault. In Cárdenas-Gallo, Sarmiento, Morales, Bolivar, and Akhavan-Tabatabaei (2017), an ensemble approach is introduced to predict the degradation of track geometry, where different models have been developed by concerning regression, deterioration, and classification. In the context of electrical systems, Le et al. (2020) introduced an ensemble fault detection approach, where they developed a boosting converter CPL and a buck converter constant power load (CPL) to study the different arc fault behaviors.

All in all, three main limitations can be observed in earlier studies: (1) most are limited to models that use only a few input parameters to model vehicle behavior, (2) high-resolution data is used in a short time to model the behaviors, and (3) they suffer from the lack of available real-life data and high prediction accuracy.

Our study addresses these issues by deploying a complex ensemble-based system on low-frequency but high-dimensional data collected from thousands of heavy-duty trucks. Furthermore, our proposed solution differs from the existing multi-task approaches such that it adapts the stacking and cosine annealing process to the ensemble structure to build meta-learning. It should be emphasized that the ensemble of different deep networks has been constructed through only one training process, which thus consumes less time and computational power than otherwise.

To our knowledge, adapting ensemble deep neural network solutions in multi-task learning problems to predict vehicle behavior in the long term is novel. The approach addresses the vehicle behavior prediction problem by low-resolution data in the automotive industry, which is generic and applicable to a variety of prediction problems in different sectors. Borrowing from the taxonomy in Zhang and Gao (2021), our vehicle behavior model can be characterized as a multi-task stacked-ensemble approach since it takes into consideration the prediction of several diverse snapshot models to construct the meta model, and then the model is used for final behavior prediction.

3. Data representation

This section describes the two data sets used for the proposed vehicle behavior modeling approach: Logged Vehicle Data (LVD), which is low-resolution (infrequent) but high-dimensional (many features) data that is aggregated in a cumulative fashion; and Dynafleet data, consisting of high-resolution (frequent) but more low-dimensional (fewer features) data.

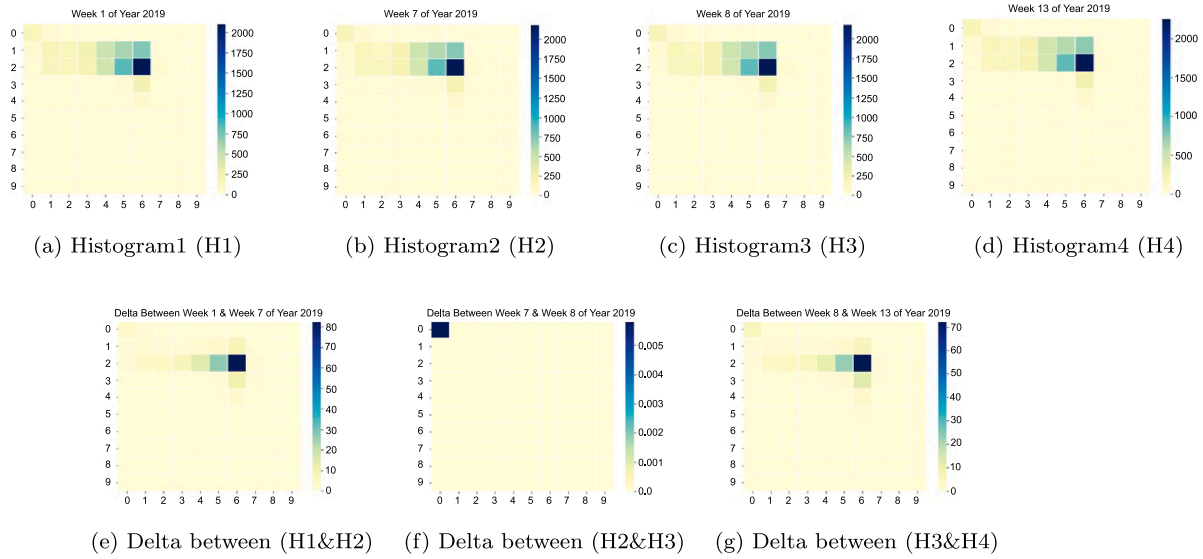
3.0.1. Low resolution data—Logged Vehicle Data (LVD)

The logged vehicle data (LVD) were collected from commercial trucks over 2019. The LVD holds the aggregated sensor information for a fleet of heavy-duty trucks operating worldwide. The values of the features are collected using telematics and through manual inspection when a vehicle visits an authorized workshop for repairs and service. In general, two types of features are logged in this dataset. The first type of feature shows the operational sensor values for the vehicle in the form of scalars and histograms during its operation. This data is continuously aggregated and includes several features such as oil temperature, fuel consumption, compressor usage, gears used, cargo load, etc. The scalars are commonly "life of vehicle" values, meaning they represent the cumulative value so far in the vehicle's life, such as the total mileage, fuel consumption or engine time. The histograms represent the total time spent in a certain bin defined by the histogram axes. For instance, the histogram describing the engine speed and engine torque has bins that contain the total time spent in a certain interval in engine speed and engine torque. The read out frequency of these parameters will vary depending on importance, from every day to occasional workshop visits. This complicates the process of calculating the deltas that describe how the vehicle has been used between readouts; how this is handled is described in Section 5. The second type of feature expresses the configuration of the vehicles, for instance, the type of engine, gearbox, pumps, and similar. This information is assumed to not change over a vehicle's life. Fig. 1 shows an example cumulative and non-stationary tabular/scalar features over one year.

3.0.2. High resolution data-dynafleet data

The Dynafleet database contains fleet management data related to the services provided by Dynafleet (Volvo Trucks) and Optifleet (Renault trucks). Depending on the services provided, the vehicles have tracking events logged with intervals of 1 to 10 min, and in connection with certain events such as turning the engine on or off. The tracking events include information such as GPS position, speed, odometer, and accumulated fuel consumption. Using this data, it is possible to estimate the average vehicle behaviors described below. The labels rely on positional data and cannot be directly computed from the aggregated data in the LVD dataset.

- **Geopattern:** Classification according the approximate size of the area traveled during the desired year. The extent of the geographical cloud of positional data during a year is used to determine the classes.
- **Geopattern repetitive:** Whether the routes traveled are mixed or repetitive.
- **Longest stop:** Classification according to the duration of the daily longest stop (typically the night stop). The classes are determined by segmenting the trucks into classes according to the yearly median of their daily longest stops.



(a) Histogram1 (H1) (b) Histogram2 (H2) (c) Histogram3 (H3) (d) Histogram4 (H4)

(e) Delta between (H1&H2) (f) Delta between (H2&H3) (g) Delta between (H3&H4)

Fig. 2. Illustration of the original and extracted histograms ($h_{i_{new}}$) from one vehicle. The subplots a–d show the LVD measurements/histograms (Coolant Temperature vs. Oil Temperature) logged in weeks 1, 7, 8 and 13; and subplots e–g indicate the extracted delta between the above weeks (I will remove the chassis id from the plots).

- **Night stop class:** Classification according to if the vehicle has a common home base or an irregular night stop pattern. The night stop locations are found by clustering the night stops. The number of resulting clusters needed to account for the majority of the stops determines the class.
- **Driver class:** Classification according to the number of regular drivers of the vehicle (one main driver or several different ones), based on the total number of unique drivers and the distance driven by each driver.

All of these labels are calculated by an internal Volvo Group framework used for vehicle utilization analysis from Dynafleet data. The labels have been used and their precision verified in several projects. The framework can output labels for different time frames but only the yearly labels are used in this study.

4. Problem formulation

This section presents the formulations determined to tackle vehicle behavior modeling. Our approach considers low-resolution data (LVD), which are labeled (Geopattern, Geopattern repetitive, Longest stop, Night stop class, and Driver class) with high-resolution data as the system's input, and employs multi-task ensemble deep neural networks. The intention with this vehicle behavior modeling and prediction investigation can be formulated as follows:

- The design of an ensemble machine learning approach is studied, formulating the task as a supervised learning problem in a multi-task fashion, to model and predict vehicle usage. Given labels determined from high-resolution data, the ensemble multi-task approach maps the usage to multiple vehicle behaviors simultaneously.

5. Methods

Fig. 3 shows the high-level structure of the proposed vehicle behavior modeling approach for the automotive industry. It contains two main parts: Data pre-processing and Multi-task Ensemble Modeling. The integration of the labels to low-resolution readouts and feature extraction processes are done in the first module. Given the low-resolution usage data, the ensemble method then builds the predictive model over two phases to categorize the vehicle's behavior. The modules and sub-modules are described below (see Fig. 4).

5.1. Data pre-processing

5.1.1. Data integration

This module merges the LVD and the labels calculated from the Dynafleet data to create an integrated dossier with the usage and behavior designations. These two sources of data are merged based on the vehicles' "Chassis id", and "Date of readout". The labels from Dynafleet data are calculated yearly and the LVD data are therefore also labeled yearly. The LVD readouts can be available weekly, which means that every vehicle has a maximum of 52 weeks of LVD data assigned with the same labels (i.e. we assume that the vehicle behavior is the same all through the year).

5.1.2. Feature engineering

This sub-module serves as an extraction component to compute derived statistics from the original histograms. These are weekly differences between histograms. In the feature engineering process, the *delta* between subsequent cells in each histogram/readouts is calculated and exploited as new features. Fig. 2a–d show different weekly histograms (Coolant Temperature vs. Oil Temperature) and Fig. 2e–g illustrated the extracted deltas between the weeks.

5.2. Multi-task learning:

Multi-task Learning (MTL) refers to the idea of learning several related tasks at the same time with the same algorithm to learn representations that are more general, lead to improved generalization, and possibly also faster learning (Zhang & Yang, 2021). In our specific case of modeling vehicle behaviors, the different behavior tasks are denoted τ_i and $\boldsymbol{\tau} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ is the set of all task. The total loss vector is:

$$min_\theta L(\theta) = (L_{\tau_1}(\theta), L_{\tau_2}(\theta), \ldots, L_{\tau_m}(\theta))^{\boldsymbol{\tau}} \quad (1)$$

where $L_{\tau_j}(\theta)$ is the loss function of the j_{th} task (each task refers to a particular behavior mentioned in Section 1). A multi-task learning aims to perform joint learning at the same time and optimize all the tasks by utilizing $D = \{x_j^i, y_j^i\}_i$ with $x_i^t \in X, y_i^t \in Y$ as data samples. The models learn from the data points D, and takes advantage of θ_s as the shared layers to calculate a loss for each task $L(\tau_i|\theta_s, \theta_i)$. Thus, lets assume, a multitask loss function parameterized by $\{\theta_s\} \cup \{\theta_i|\tau_i \in \boldsymbol{\tau}\}$, where "$\theta_s$" indicates the shared parameters in the shared layers and τ_i is the task i with specific parameters. Thus, given a batch of samples X, the total

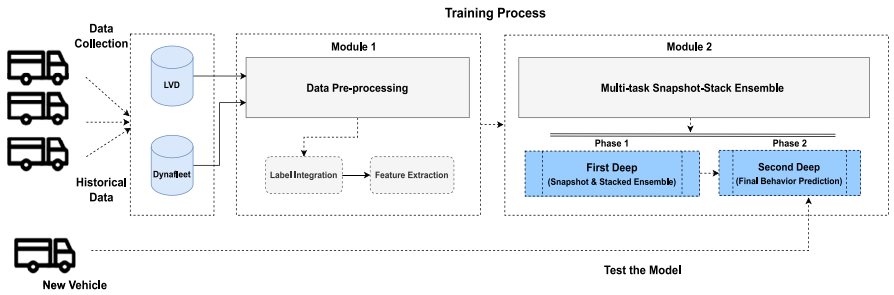


Fig. 3. The high-level conceptual view of the proposed approach.

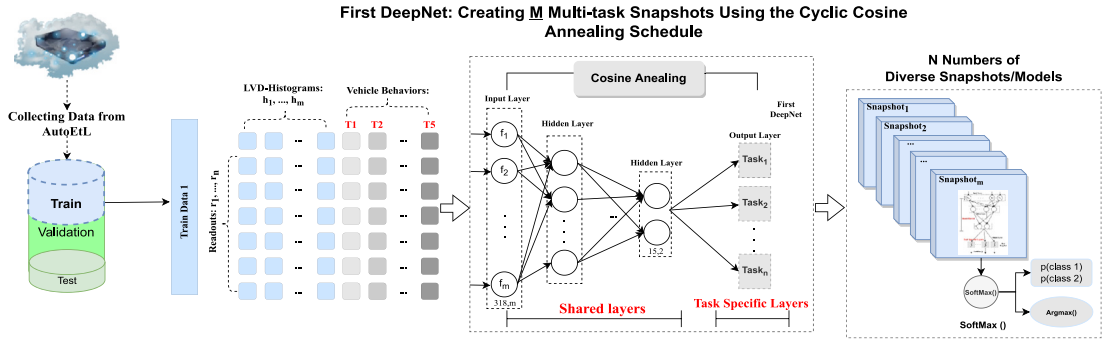


Fig. 4. A conceptual view of the proposed multi-task ensemble approach (first phase). In this phase, the LVD data (injected from Module 1) is partitioned into three parts: train, validation, and test. The training and validation data are utilized in the first phase to generate the snapshot (SPs) models. In this process, CCSA is done to construct the snapshots where the function periodically decreases the learning rate from α_0 to ≈ 0 in a cycle—in this study, we parameterized the number of snapshots to obtain the optimal numbers – (see Fig. 6 for visualization). At the end of each cycle, the model with the best performance is saved as the snapshot.

loss function of the MTL for vehicle behavior prediction is calculated by Eq. (2).

$$L_{all}(X, \theta_s, \theta_i) = \sum_{k=1}^{m} L(\theta_k) \tag{2}$$

5.3. Multi-task training using snapshot-stacked ensemble

The multi-task ensemble approach contains two main parts. In the first part, a multi-task deep neural network is created to generate several snapshot models from a single training process. The snapshots are then used to create a new data set using the validation set. In this practice, we utilized the histograms dedicated for validation to test the snapshot models. The second part is designed for building a meta-model, where the outcome of each snapshot is horizontally added to the validation set for the final multi-task training and prediction. Indeed, the meta-learning network attempts to learn from the output of multiple snapshots to decrease the prediction errors obtained from the first deep neural network.

5.3.1. Ensemble on multi-task snapshots outputs

The ensemble technique is a procedure in which several different predictive models are developed to lower the prediction variance. If the models are different but with similar bias, then combining them should lead to lower variance and thus also lower prediction error. Training multiple deep neural networks is expensive in computing resources so the Snapshot Ensemble is used to build multiple multi-task models by a single training process. However, this method suffers from often generating very similar models which diminishes the value of using an ensemble (Huang, Li, Pleiss, Liu, Hopcroft et al., 2017). We overcome this by imposing more diversity among the individual models by changing the learning rates to construct diverse networks over the course of training. To this end, the Cyclic Cosine Annealing Schedule (Fu et al., 2019) is used to split the training process into N cycles, each initiated with a large learning rate that relatively rapidly drops to the minimum value before increasing quickly again. Eq. (3) shows the learning rate over the iterations.

$$\alpha(t) = \frac{\alpha_0}{2}\left(cos\left(\frac{\pi \; mod(t-1, \lceil T/n \rceil)}{\lceil T/N \rceil}\right) + 1\right) \tag{3}$$

where, $\alpha(t)$ is the learning rate at iteration t, α_0 is the initial learning rate at the beginning of each cycle, and T is the total number of epochs over training. Hence, Eq. (3) anneals the learning rate from α_0 to $f(T/C)$ which is ≈ 0 over the course of a cycle (Huang et al., 2017). N is the number of cycles that the network is trained throughout the training process, in which the best weights at the end of each cycle are saved as the snapshot of the model. This eventually leads to N multi-task snapshot models. Indeed, in this approach, the learning rate increased periodically, which supports the model to converge at the global minimum rather than local minimum during the snapshot ensemble. Furthermore, it avoids having to manually find the optimal maximum learning rate. The overall snapshot process with multiple tasks is illustrated in Fig. 6, where at the end of each cycle the network with the maximum performance (here we consider the overall performance form all the tasks) is saved as the snapshot. All the generated models are then utilized to forecast vehicle multiple behavior in the first phase.

5.3.2. Training the multi-task network

Table 2 shows the construction of the network in the first phase, where the network is incrementally trained, creating a Sequential model. At the beginning of the network, there are several shared layers for all the tasks. Each task is then allocated its own layer as the task-specific layer. We have used the modified version of the "Xavier initialization" (Kumar, 2017) called "He" to initialize the network parameters (Arpit & Bengio, 2019; He, Zhang, Ren, & Sun, 2015) (shown in Eq. (4)), where G is the Gaussian probability distribution with zero mean and a standard deviation of $\sqrt{\frac{2}{n_l}}$. In this initialization form, biases are initialized at 0 and n_l is the number of inputs to that

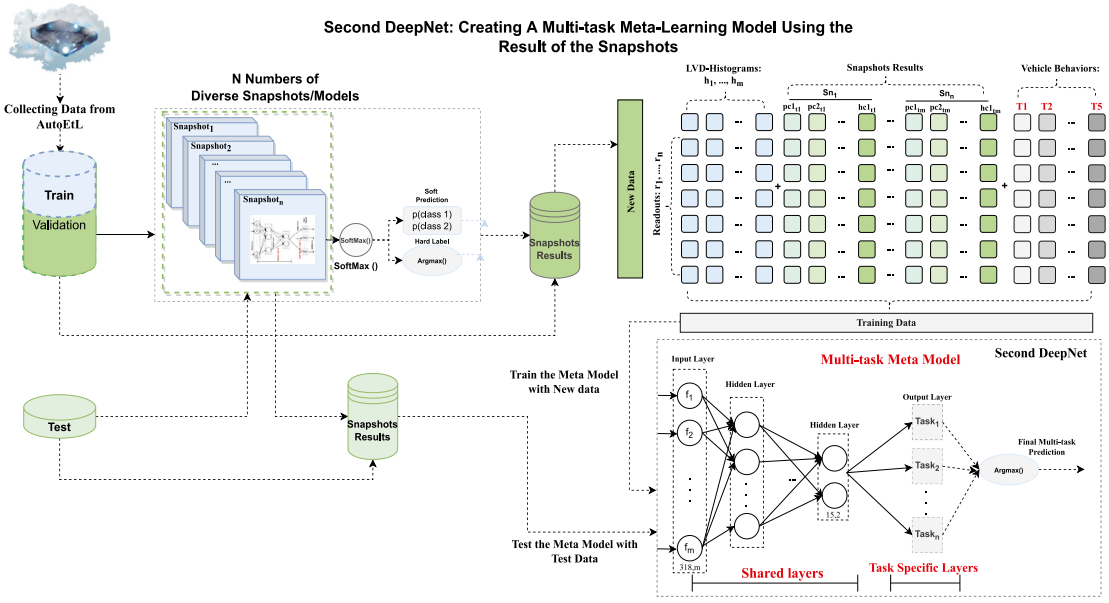


Fig. 5. A conceptual view of the second phase of the proposed system. In this phase, the generated outputs are horizontally added to the original data set (validation data) to induce a new dataset. Accordingly, the new dataset is used to build the meta-learning for the final prediction. Finally, the test part, which was held out, has been utilized (with the new dimensions) to validate the meta-learning performance.

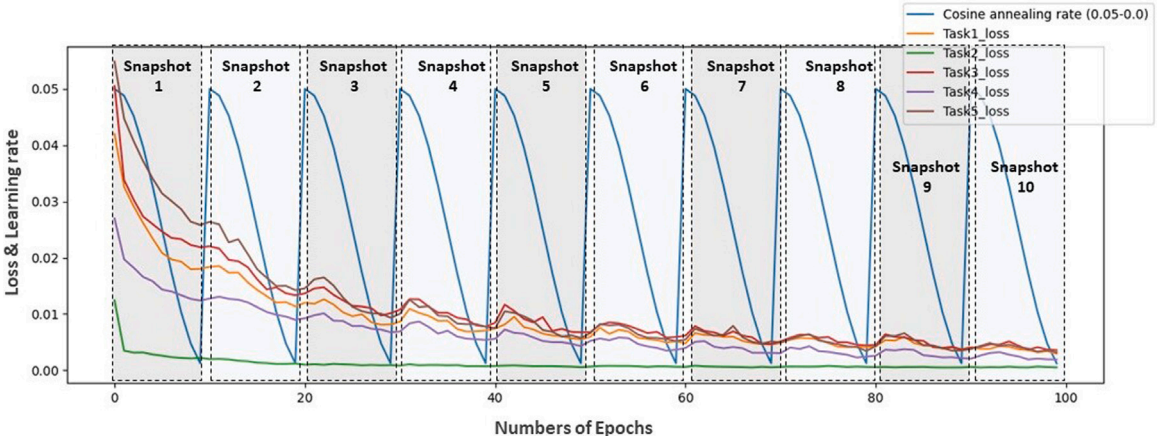


Fig. 6. Visualization of different tasks losses and cosine annealing learning and cycles (blue). In this practice, $N = 10$ snapshot models are generated and saved to be used in the first and the second phase of the multi-task networks.

node in layer l.

$$w_l = G(0, \sqrt{\frac{2}{n_l}}) \tag{4}$$

In order to use stochastic gradient descent to train the deep neural networks, rectified linear unit (ReLU) (Ramachandran, Zoph, & Le, 2017) as the – active function – is utilized to transfer the summed weighted input from the nodes into the output of the node.

$$f(x) = max(o, x) \tag{5}$$

In Eq. (5), $f(x)$ is the $ReLU$ function, where it outputs 0 for $\forall\ x < 0$, while for positive inputs $x \geq 0$, it returns the same value $f(x) = x$. The Batch Normalization (Ioffe & Szegedy, 2015) technique is used after each fully connected dense layer – in the shared and specific layers – to normalize the input of each layer to reduce the internal covariate shift problem leading to stabilizing the learning process. This is followed by using the Stochastic Gradient Descent (SGD), which is an optimization technique mainly utilized in neural network predictive models to find the parameters which correspond to the best fit between predicted and actual outputs. The technique executes periodic updates with high variance, enabling the objective function to fluctuate laboriously and leap to new and potentially better local minima (Ruder, 2016). We "Dropout" is used to reduce the risk for over-fitting, both in the shared and specific layers. Basically, this led to providing different multi-task neural network architectures when training the networks. In each specific output layer, we employed the Soft-max activation function – depicted in Eq. (6) – for the probability distribution of the target variables. In our multi-task problem every task has different classes, so we are dealing with a multi-task multi-class problem in this study.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K_{\tau_i}} e^{z_j}} \tag{6}$$

where $\sigma(z_i)$ are the elements of the input vector to the softmax function, K is the number of classes in task τ_i and $\sum_{j=1}^{K_{\tau_i}} e^{z_j}$ is the normalization term, which ensures all the output values of the function will sum to 1 (each class probability should be in the range [0, 1]). Finally, the output of Eq. (6) will be injected to the $Argmax$ function (Agarap, 2018) – defined in Eq. (7) – to predict the hard label as a class of the vehicle

Algorithm 1: The proposed MLTS snapshot ensemble approach

Input: R ; // readouts $R = \{r_1, \dots, r_n\}$
Output: $Fp_{\tau_{i,\dots,m} \in \tau}$; // final multi-task predictions
$Trd \in R$; // train data
$Vld \in R$; // validation data
$Tsd \in R$; // testing data
Sn ; // Snapshot
$Ls \leftarrow \{\}$; // list of Snapshots
$Dt \leftarrow \{\}$; // New dataset
$CCAS$; // Cosine Annealing function (Equation (3))
N ; // number of cycles, (Equation (3))
$Deepnet1$; // 1st multi-task net to generate snapshots (Table 2)
$Deepnet2$; // 2nd multi-task net (meta-learning) for final prediction (Table 3)
// Phase 1 is started:
$Train\ (Deepnet1\ with\ CCAS|Trd)$ **while** $i \leq E$ **do**
  if $mod\ (i \backslash N) == 0$ **then**
    $Ls \leftarrow Sn$; // save the snapshot in Ls
  else
// Creating a new data set
$Load\ Ls$; // load all snapshots
for $S_i \in Ls$ **do**
  for $\tau_i \in \tau$ **do**
    $p = S_i.predict(Vld_{\tau_i})$; // validate S_i with validation set
// phase 2 is started:
// Creating a new data set
for $S_i \in Ls$ **do**
  $Dtr \leftarrow stack(p, Vld)$; // stack the prediction to Vld
for $S_i \in Ls$ **do**
  $Dts \leftarrow stack(p, Tsd)$; // stack the prediction to Tsd
$Mm \leftarrow Train\ (Deep2|Dtr)$; // building a meta learning
$Fp \leftarrow Mm.predict(Dts)$; // final prediction

behavior.

$$hp_{r_i} = argmax(\sigma(z_i)) \tag{7}$$

where hp_{r_i} carries the hard label prediction of the readout r_i in the test set, expressing whether the usage corresponds to a certain vehicle behavior.

5.3.3. Multi-task stacking and ensemble

Each snapshot multi-task model Sn provides two sorts of outputs such as; Soft prediction (Eq. (6)) and Hard label (Eq. (7)). The former returns the predicted probability of the classes belonging to the specific task for the given vehicle usage. The second output is the hard label representing the predicted behavior for each task. Accordingly Eq. (8) shows the numbers of the new features which will be generated by the Snapshots models.

$$feature_{new} = N_{Sn} \times \sum_{k=1}^{m} (C_{\tau_k} + 1) \tag{8}$$

where N_{S_n} refers to the snapshot numbers, and m is the number of tasks τ. C_{τ_k} points to the multiple class of the vehicle behaviors which is summed by "1" as the hard-label of that task. Table 1 illustrates the multi-task snapshot and stacking horizontal ensembles calculations, which are described in detail below:

- $r_1, \dots, r_n$: refer to the vehicle non-stationary readouts/histograms, which are collected over time,
- $h_1, \dots, h_m$: represent the usage measurements/histograms characterizing a vehicle's behavior in r_i,
- $P_{r_1(Sn_1)} = (p_{c_1}, \dots, p_{c_k}, hp_{r_1})_{\tau_1,\dots,\tau_n}$, indicate the output of the multi-task snapshot models–in the first phase–for the given readouts by Sn_1. Here, each Snapshot model sp_i generates multiple probabilities for each task,
- $p_{c_1}, \dots, p_{c_k}$, represent the predicted probability of different classes of a vehicle behavior for the given readout
- hp_{r_i} points to the hard prediction of the specific task for the given readout

The output of the soft predictions and the hard labels for each specific task τ_i are horizontally combined with the original features (validation set, see Fig. 5). This means each snapshot model could add more (calculated by Eq. (8)) parameters carrying the knowledge of the previous models' performance on that specific readout. The new data is then passed to the second deepnet – second phase – to build the meta-model to reduce the multi-task forecasting errors obtained from the previous phase. Finally, the performance of the meta-model will be assessed by the test set.

5.4. Task transference

To gain insights on the impact of the tasks on each-other during the ensemble training, we adapt the inter-task affinity to vehicle behavior of meta-learning. Thus, considering a meta learner that is parameterized by θ, at the time t of training the vehicles behavior in the second deepnet, given a batch of $X_t \in D$. We define the quantity $\theta^{t+1}_{s|i}$ to indicate the model with the updated shared parameters towards the task τ_i.

$$\theta^{t+1}_{s|i} = \theta^t_s - \zeta \Delta_{\theta^t_s} L_{\tau_i}(X, \theta_s, \theta_{\tau_i}) \tag{9}$$

Using $\theta^{t+1}_{s|i}$ in Eq. (9), we could measure the impact of task τ_i on the performance of the other tasks defined in $\tau = \{\tau_1, \tau_2, \dots, \tau_m\}$. Thus, given the input $X_t \in D$ we can measure the loss for each task taking the updated shared parameter θ_s as well as the specific task parameters θ_i. Indeed, we assess the impact IM of gradient update of task τ_i on a given task τ_j, so then we can compare the ration between loss value of task τ_j before and after conducting the gradient update from task τ_i towards the shared parameters as follows:

$$IM_{\tau_i \to \tau_j} = 1 - \frac{L_{\tau_j}(X^t, \theta^{t+1}_{s|i}, \theta_j)}{L_{\tau_j}(X^t, \theta^t_s, \theta_j)} \cdot W_c \tag{10}$$

$$W_c = \frac{t}{T} \quad t = 1, 2, \dots, T \tag{11}$$

Thus, we translate $IM_{\tau_i \to \tau_j}$ as a measure of transference from meta-train task τ_i to task τ_j. The positive the value of $IM_{\tau_i \to \tau_j}$ shows the update on the shard parameters θ_s led to in a lower loss value on task τ_j with respect to the original parameters. This basically expresses the positive effect of task i to generalize the predictive model on task τ_j, while the negative value of $IM_{\tau_i \to \tau_j}$ describes the destructive impact of task i on task τ_j over the training process. We adapt the overall inter-task affinity measure idea (Fifty, Amid, Zhao, Yu, Anil et al., 2021) and improved it by incrementally adding certain weights, defined in Eq. (11) on each iteration over the training process. t refers to the current epoch number and T is the maximum number of epochs that the multi-task network should be iterated. This is due to the fact that at the beginning of the training, the weights are randomly generated, so it is not expected the earlier loss have the same impact w.r.t to the parameters at the end of the training. Thus, in Eq. (12) we calculate the overall transference from task i to other tasks:

$$\hat{IM}_{\tau_i \to \tau_j} = \frac{1}{T} \sum_{j=1}^{m} IM_{\tau_i \to \tau_j} \tag{12}$$

$\hat{IM}_{i \to j}$ can be employed in different levels of granularity such as per-epoch level or even micro-level/batch level. In this study, we measure the transference at epoch level.

Table 1
The illustration of the snapshots multi-task outputs and stacked ensemble.

Readouts	Original features			Snapshot models' predictions		
	h_1	...	h_m	$P_{r_1(Sn_1)} = (p_{c1}, \dots, p_{ck}, hp_{r_1})_{\tau_1,\dots,\tau_n}$, $P_{r_1(Sn_2)} = (p_{c1}, \dots, p_{cl}, hp_{r_1})_{\tau_1,\dots,\tau_n}$	...	$P_{r_1(Sn_m)} = (p_{c1}, p_{c2}, hp_{r_1})_{\tau_1,\dots,\tau_n}$
r_1	$r_{1_{h_1}}$	...	$r_{1_{h_m}}$			
r_2	.	...				.
.	.	...				.
.	.	...				.
r_n	$r_{n_{h_1}}$	...	$r_{n_{h_m}}$	$P_{r_n(Sn_1)} = (p_{c1}, \dots, p_{ck}, hp_{r_n})_{\tau_1,\dots,\tau_n}$, $P_{r_n(Sn_2)} = (p_{c1}, \dots, p_{cl}, hp_{r_n})_{\tau_1,\dots,\tau_n}$	...	$P_{r_n(Sn_m)} = (p_{c1}, p_{c2}, hp_{r_n})_{\tau_1,\dots,\tau_n}$

Table 2
The architecture of the first multi-task deep networks (Deep1).

Type of layers in multi-task	Layer	Output shape	# of parameters
Shared Layers	layer00 (Dense)	(None, 272)	74256
	batch00 (Batch Normalization)	(None, 272)	1088
	dropout (Dropout)	(None, 272)	0
	layer10 (Dense)	(None, 68)	18564
	batch01 (BatchNormalization)	(None, 68)	272
	layer15	(None, 34)	2346
	dropout (Dropout)	(None, 34)	0
	layer20 (Dense)	(None, 22)	770
	batch01 (BatchNormalization)	(None, 22)	88
	layer30(Dense)	(None, 13)	299
Specific Layer for Task 1	layer31(Dense)	(None, 7)	220
	batch02 (BatchNormalization)	(None , 7)	120
	Target 1 (Dense) Softmax()	(None, 3)	42
Specific Layer for Task 2	layer32(Dense)	(None, 7)	220
	batch03 (BatchNormalization)	(None, 7)	78
	Target 2 (Dense) Softmax()	(None, 2)	28
Specific Layer for Task 3	layer33(Dense)	(None, 7)	220
	batch04 (BatchNormalization)	(None, 7)	78
	Target3 (Dense) Softmax()	(None, 4)	56
Specific Layer for Task 4	layer34 (Dense)	(None, 7)	78
	batch05 (BatchNormalization)	(None, 7)	78
	Target4 (Dense) Softmax()	(None, 2)	28
Specific Layer for Task 5	layer35 (Dense)	(None, 7)	78
	batch03 (BatchNormalization)	(None, 7)	78
	Target 5 (Dense, 3)	(None, 3)	42
Total params: 172,323 Trainable params: 171,505 Non-trainable params: 818			

5.5. Task grouping

Although in our multi-task formulation we aimed to optimize the performance of all the tasks', due to the nature of the tasks or may because of the dissimilarity between them, it is quite challenging to reach the optimal outcome simultaneously. Thus, we took advantage of $I\hat{M}_{\tau_i \to \tau_j}$ to identify certain tasks that may perform better if they are trained together in the multi-task formulation. Hence, mathematically, given a set of tasks $\boldsymbol{\tau} = \{\tau_1, \tau_2, \dots, \tau_m\}$, we aim to construct a group of k multi-task ensemble deep networks $M = \{m_1, m_2, \dots, m_k\}$ such that $\forall \tau_i \in \tau$. m_j is the j_{th} multi-task ensemble network which considers $X \in D$ as an input, and simultaneously trains a set of tasks $\{\tau_2, \tau_5, \dots, \tau_p\} \in \boldsymbol{\tau}$, while serve only subset of the tasks for the prediction. This means with m numbers of tasks in $\boldsymbol{\tau}$, there will be $2^m - 1$ non-empty subsets of the tasks $\{1, \dots, n\}$. Thus, given the predictive performance $_F$ from the second deepnet, we therefore define the aggregate performance of our task grouping as $\sum_{i=1}^{n} {}_F(\tau_i|M)$ where ${}_F(\tau_i|M)$ indicates the performance of task τ_i from the set of models M exploiting the model m_j which the task grouping algorithm predicts the performance of task τ_i will be highest.

6. Experimental evaluation and results

The results and evaluations are presented with respect to the three research questions introduced in Section 1. RQ1: To what extent could the behavior of vehicles be modeled and predicted from the low frequency, but high-dimensional data using the multi-task ensemble deep neural approach? RQ2: To what extent could the vehicle behavior associated with one task affect the prediction performance of the other tasks over the training process? RQ3: Will vehicle behavior grouping enhance the performance of the predictive multi-task model?

Table 3
The architecture of the second multi-task deep networks (Deep2).

Type of layers in multi-task	Layer	Output shape	# of parameters
Shared Layers	layer00 (Dense)	(None, 462)	213906
	batch00 (Batch Normalization)	(None, 462)	1848
	dropout (Dropout)	(None, 462)	0
	layer10 (Dense)	(None, 115)	53245
	batch01 (BatchNormalization)	(None, 115)	460
	layer15	(None, 57)	6612
	dropout (Dropout)	(None, 57)	0
	layer20 (Dense)	(None, 38)	2204
	layer30(Dense)	(None, 30)	1170
Specific Layer for Task 1	layer31(Dense)	(None, 30)	930
	batch02 (BatchNormalization)	(None, 30)	120
	Target 1 (Dense) Softmax()	(None, 3)	93
Specific Layer for Task 2	layer32(Dense)	(None, 30)	930
	batch03 (BatchNormalization)	(None, 30)	120
	Target 2 (Dense) Softmax()	(None, 2)	62
Specific Layer for Task 3	layer33(Dense)	(None, 30)	930
	batch04 (BatchNormalization)	(None, 30)	120
	Target 3 (Dense) Softmax()	(None, 4)	124
Specific Layer for Task 4	layer34 (Dense)	(None, 30)	930
	batch05 (BatchNormalization)	(None, 30)	120
	Target 4 (Dense) Softmax()	(None, 2)	62
Specific Layer for Task 5	layer35 (Dense)	(None, 30)	930
	batch03 (BatchNormalization)	(None, 30)	120
	Target 5 (Dense, 3)	(None, 3)	93
Total params: 499,295 Trainable params: 497,711 Non-trainable params: 1,584			

6.1. RQ1: Vehicle behavior modeling results

Several lines of evaluations were performed. It was explored how the number of snapshots affected the results. The number of cycles N (see Eq. (3)) was varied $N = \{5, 10, 15, 20\}$, which generated multiple diverse snapshot models over the 200 epochs. The diversity within the resulting snapshot ensemble was then estimated in the following way: Let $X = \{x_n, y_n\}_{n=1}^{N} \in D$ represent the vehicle usage, where $y_n \in \{1, \ldots, J\}$ is the true label out of J different vehicle behaviors for the input x_n, then the diversity between the snapshots with the disagreement measure defined in Eq. (13) (Heidemann, Schwaiger, & Roscher, 2021).

$$dis_{i,j}^{t_l} = \frac{1}{N}\sum_{k=1}^{N} fn(Sn_i(r_k)^{t_l} \neq Sn_j(r_k)^{t_l}) \tag{13}$$

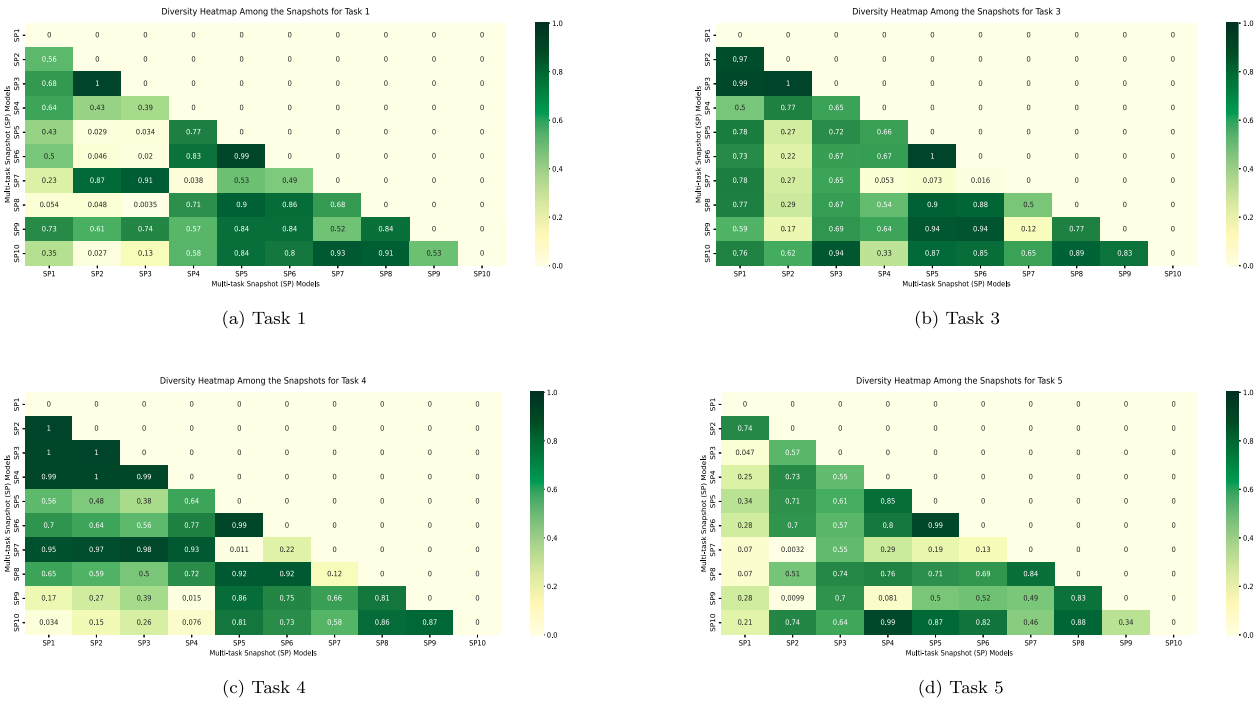
Here $Sn_i(r_k)$ refers to the label assigned by the snapshot (Sn_i) to readout r_k for the task $\tau_l \in \mathcal{T}$. $fn()$ is the function of truth predicate that counts the cases where Sn_i was correct and the Sn_j was wrong and vice versa. Fig. 7 illustrates the disagreement between the 10 snapshots. The heat-map shows the diversity between the snapshots, where the closer the dis-value is to 0 the more diversity there is between the models.

In all experiments were 60% of the LVD data used to train the snapshots and 30% were used as validation set to obtain the soft predictions and hard labels. Finally, 10% were held out for testing the meta-learning model in the final stage.

The bar plot in Fig. 8 shows the ensemble networks' performance on different tasks, when trained with varying numbers of snapshots. The f-score[1] metric is used to assess the performance of the networks, such that all the networks with different snapshots are constructed five times to obtain the overall performance. The plot shows that ensemble networks with 15 snapshots are consistently better than those with 5, 10, and 20 snapshots. The figures show 0.89 for Task 1, 0.99 for Task 2, 0.88 for Task 3, 0.76 for Task 4 and finally 0.87 for Task 5. The low standard deviations from all the tasks compared to the other networks indicate stability of the ensemble network with 15 snapshots. It should be noted that Task 2 (geopattern repetitive) is highly imbalanced since trucks in the data set rarely only have a few distinctive routes during the year.

In the remainder of the experiments, 15 snapshots were used throughout. In the next evaluation it was tested how the performance changed with single- versus multi-task learning. Both the single- and multi-task learning were tried with different ways to form ensembles, e.g., Boosting, Bagging, or Stacking. The aim was to both have different baselines that could be compared with our finally proposed system, and to see whether the tasks can be modeled and predicted better through the single- or multi-task formulation. Regardless of the computational benefit of multi-task learning, this also addresses the question *whether the knowledge within different tasks can positively affect the predictive performance compared to the single training?*

[1] The sklearn library was used to measure the f-score for multi-class classification:https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.



(a) Task 1 (b) Task 3

(c) Task 4 (d) Task 5

Fig. 7. The disagreement between 10 snapshot models generated by cosine annealing for the multi-task problem. The heat-map shows the diversity in the generated snapshots in the first phase of the multi-task ensemble approach. In these matrices, each cell under the diagonal represents the disagreement between two snapshots – e.g., Sn_1 vs. Sn_2–. The closer the dis-value (Eq. (13)) is to 0, the more diverse the models are. The heatmap shows the snapshots for Task 1, Task 3, Task 4 and Task 5. Since Task 2 is highly imbalanced and the disagreement is calculated based on the output of the snapshots, we do not plot Task 2.

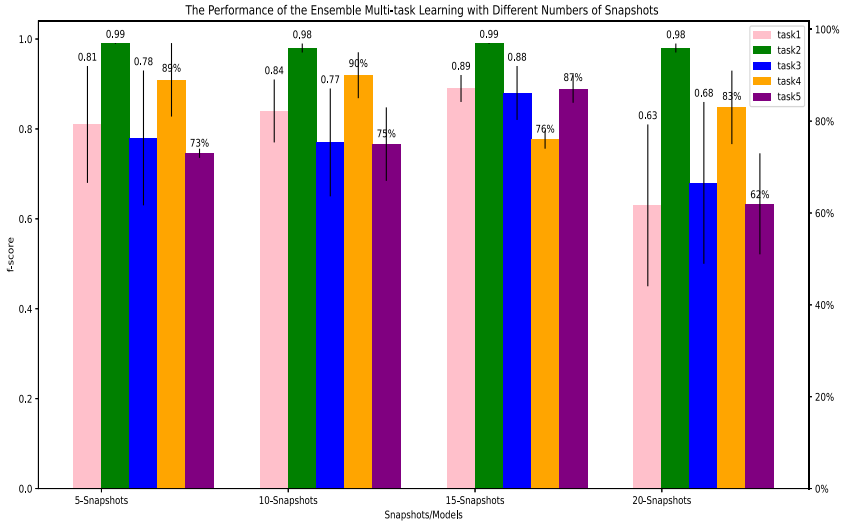


Fig. 8. The performance of the MTSSE with different numbers of snapshots.

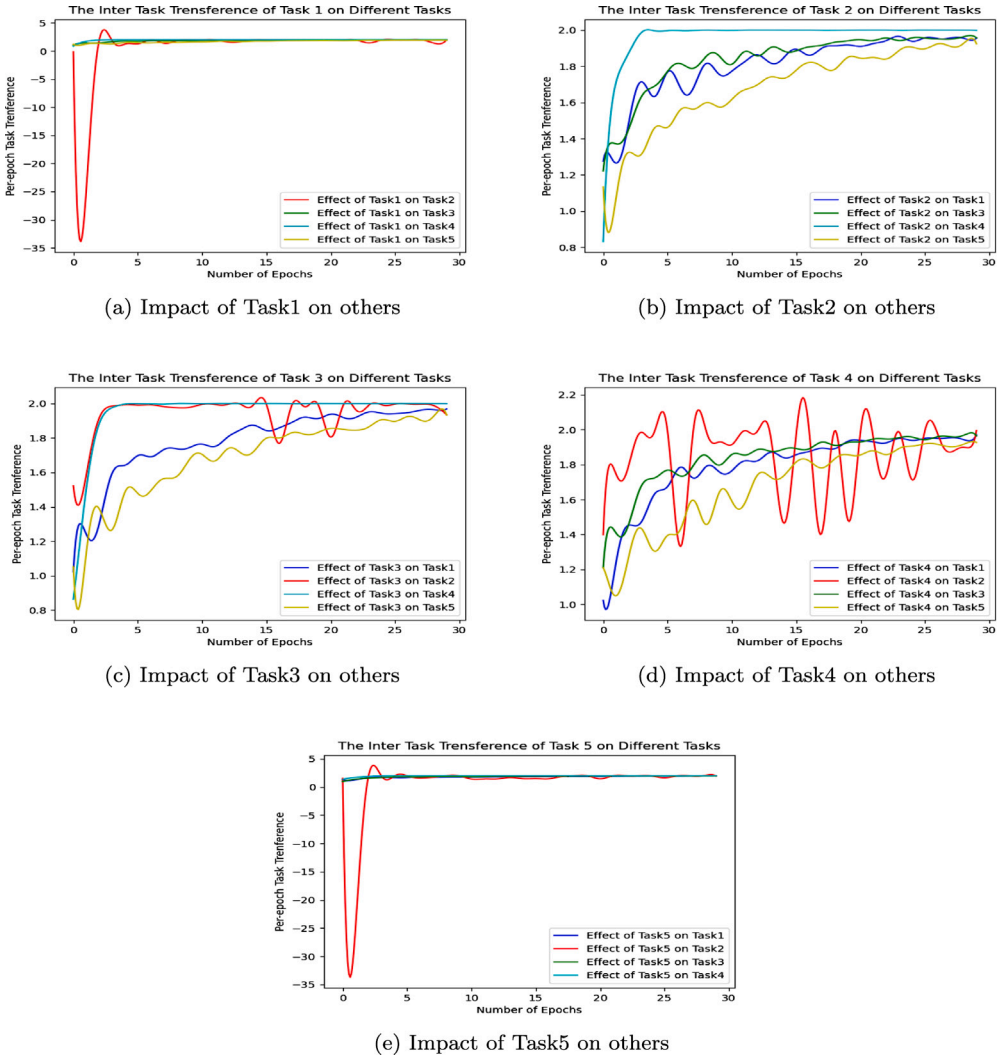
Table 4 lists the results of the single- and multi-task learning experiments. Looking only at the single-task formulation, the figures obtained from the ensemble-based approaches such as Boosting (XGB, LGB, CatBoost, AdaBoost), Bagging, and Stacking illustrate how the vehicle behavior modeling is a complex task, in particular Task 1, Task 3, and Task 5. By ignoring the high f-score values achieved for Task 2, due to the high-imbalance of the classes, in most cases the approaches performed poorly with f-scores below 70%. The proposed ensemble snapshot approach as a single task performed well in all tasks except for Task 3, achieving 0.91 for Task 1, 0.73 for Task 4, and 0.79 for Task 5. Task 5, the single or multiple driver class, is quite a difficult task where other approaches performed poorly. Compared with other ensemble-based approaches, STSSE provided 30% higher f-score, which is a significant improvement. A similar improvement was achieved in Task 1 by 0.91, which shows the stability of the ensemble snapshot deep neural network approach.

Table 4 shows that the proposed approach MTSSE outperformed all approaches except one, both in single and multi-task formulations. The only exception is single-task with STSSE on Task 1 (boldfaced in the table). The high f-score values (0.89 in Task 1, 0.89 in Task 3, 0.74, and 0.88 for Task 4 and Task 5, respectively) obtained with MTSSE (phase 2) indicate that deep neural networks with ensemble formulation can significantly improve the predictive performance in complex problems with multi-task learning.

The statistical t-test between MTSSE (phase 2) and other approaches in single and multi-task formulations show a significant difference between the performances in Task 1, Task 3, and Task 5. However, for Task 4 (night stop class) the difference is mostly not significant.

Table 4
The comparison between the Multi-task Snapshot Ensemble (MTSSE), Single-task Snapshot Ensemble (STSSE) and different ensemble-based classifiers for RQ1. For each model, 5 × 2 cv paired t-test was used to test the pairwise significance between the model and the MTSSE (phase 2) model. "**" refers to the alpha level at 0.05 to reject the null hypothesis, i.e. the "two sigma" level. Significant differences are denoted by ✓and insignificant differences are denoted by ✗. "MTSSE only prediction" refers to the ensemble networks constructed based only on the prediction of the snapshots, not the combination of original and predictions.

Methods	Task 1		Task 2		Task 3		Task 4		Task 5	
	f-score	t-test **	f-score	t-test **	f-score	t-test **	f-score	t-test **	f-score	t-test **
Single Task Boosting (XGB)	0.68 ± 0.04	9.55 ✓	0.97 ± 0.02	1.11 ✗	0.69 ± 0.06	4.51 ✓	0.72 ± 0.02	0.89 ✗	0.49 ± 0.03	17.1 ✓
Single Task Boosting (LGB)	0.66 ± 0.06	8.48 ✓	0.97 ± 0.01	2.06 ✓	0.62 ± 0.05	6.89 ✓	0.73 ± 0.01	1.19 ✗	0.45 ± 0.03	17.4 ✓
Single Task–Boosting (CatBoost)	0.69 ± 0.06	10.42 ✓	0.98 ± 0.01	0.10 ✗	0.64 ± 0.05	6.35 ✓	0.75 ± 0.01	−0.36 ✗	0.50 ± 0.04	17.7 ✓
Single Task–Boosting (AdaBoost)	0.59 ± 0.04	12.57 ✓	0.97 ± 0.01	2.44 ✓	0.61 ± 0.01	6.22 ✓	0.72 ± 0.02	1.00 ✗	0.49 ± 0.03	17.9 ✓
Single Task-Bagging	0.67 ± 0.04	09.96 ✓	0.98 ± 0.01	0.30 ✗	0.73 ± 0.02	4.86 ✓	0.74 ± 0.03	0.31 ✗	0.50 ± 0.01	19.8 ✓
Single Task-Stacking	0.16 ± 0.04	31.38 ✓	0.97 ± 0.01	2.27 ✓	0.61 ± 0.03	8.46 ✓	0.29 ± 0.08	9.77 ✓	0.41 ± 0.06	18.1 ✓
Single Task-STSSE	**0.91 ± 0.02**	**−1.20** ✗	0.98 ± 0.01	0.00 ✗	0.64 ± 0.04	6.49 ✓	0.73 ± 0.04	0.51 ✗	0.79 ± 0.05	2.49 ✓
Multi Task-ANN	0.64 ± 0.03	14.26 ✓	0.98 ± 0.01	0.57 ✗	0.62 ± 0.05	6.96 ✓	0.73 ± 0.06	0.32 ✗	0.52 ± 0.03	14.8 ✓
Multi Task-MTSSE phase 1	0.67 ± 0.02	13.90 ✓	0.98 ± 0.01	1.09 ✗	0.64 ± 0.04	5.61 ✓	0.73 ± 0.05	0.52 ✗	0.46 ± 0.05	14.1 ✓
Multi Task-MTSSE only (avg indv)	0.49 ± 0.12	06.84 ✓	0.98 ± 0.001	0.00 ✗	0.33 ± 0.15	6.95 ✓	0.62 ± 0.06	3.28 ✓	0.33 ± 0.07	14.0 ✓
Multi Task-MTSSE phase 2 (prediction only)	0.61 ± 0.04	14.01 ✓	0.98 ± 0.01	0.30 ✗	0.62 ± 0.06	6.61 ✓	0.22 ± 0.07	12.5 ✓	0.48 ± 0.07	9.85 ✓
Multi Task-MTSSE phase 2	0.89 ± 0.03	–	0.98 ± 0.04	–	0.89 ± 0.06	–	0.74 ± 0.04	–	0.88 ± 0.04	–



(a) Impact of Task1 on others

(b) Impact of Task2 on others

(c) Impact of Task3 on others

(d) Impact of Task4 on others

(e) Impact of Task5 on others

Fig. 9. The task transference of different vehicle behavior on each other over the training process.

Table 5
Details about the 13 datasets.

Database	# of Samples	# of Features	# of Classes
VBS-fault	8 712	4	2
Car insurance claim (CIC)	10 302	27	2
Covtype	581 012	55	3
CMIYC	27 297	22	2
KDD	494 021	42	17
Building permit (BPRM)	198 900	43	8
Diabetes	768	9	2
MNIST	9 271	787	10
Musk	6 598	168	2
Parkinson	756	757	2
Spam	4 400	59	2
Waveform	5 000	41	3
Australia	142 193	24	2

The figures shown in Table 4, in particular the comparison between single- and multi-task experiments, suggest that it is not a guarantee for success to formulate the vehicle behavior as multi-task learning. For example, STSSE provided a 0.91 f-score value vs. MTSSE (phase 2) 0.89 at predicting Task 1. This means that training Task 1 in a single-task formulation yielded a better generalization than multi-task training, which suggests that at least one of the tasks could be destructive (or less constructive) to the performance of this task. Thus, to understand when multi-task learning is beneficial, there is a need to identify a set of tasks that are not antagonistic to each other and should be trained together to increase the predictive model performance (this experiment is reported in Section 6.2.1).

6.1.1. Generality of the evaluation results

The results reported above practices directed us to also study these approaches in a different context. We conducted the proposed approach with three ensemble-based approaches on several different datasets to evaluate the generality and see whether the proposed ensemble system performs equally or better in other application domains. Note: Due to the lack of multi-target values of the available public datasets, we analyze the single-task version of the approach (STSSE) in this evaluation. However, STSSE still retains its ensemble functionality to handle the prediction practice. This is an essential consideration that assesses the generality of the approach to deal with the data coming from different contexts. The detailed information of the datasets are reported in Table 5.

Table 6 shows the results of the generality experiments, carried out on 13 different datasets. The results demonstrates that STSSE, in most cases, performed better than other ensemble approaches. Individual comparisons between STSSE and Bagging or Boosting show that STSSE almost outperforms the two approaches on 8 out of 13 datasets. A particular exception is dataset 8, where Boosting and Stacking gave significantly better results. Bagging, Boosting, and Stacking also achieve a higher accuracy on datasets 3, 7, and 11, but here the differences are not statistically significant at $\alpha = 0.05$ (**). Concerning the same comparison between STSSE and Bagging, we could observe that the p-value from the t-test on the figures obtained on datasets 1, 2, 3, 4, 5, 6, 8, 10, 13 are less than the critical value to reject the null hypothesis expressing the differences are statistically significant and conclude the STSSE outperforms the Bagging approach. It is also fair to remark that the statistical test on the results, where the STSSE outperformed Boosting and Stacking on datasets 5, 9, 12, and 13, showed that the difference between the figures is not statistically significant.

6.2. RQ2: Vehicle behavior transference results

Here, the point was to see the positive or negative impact of tasks on each other during the training process. Eq. (10) was changed to the following formula to measure the impacts:

$$IM_{\tau_i \to \tau_j} = \frac{L_{\tau_j}(X^t_{val}, X^t_{tr}, \theta^{t+1}_{s|i}, \theta_j) - L_{\tau_j}(X^t_{val}, X^t_{tr}, \theta^t_s, \theta_j)}{L_{\tau_j}(X^t_{val}, X^t_{tr}, \theta^t_s, \theta_j)} \quad (14)$$

where, X^t_{val} and X^t_{tr} are the independent vehicles usage predictors for training and validating, respectively. $L_{\tau_j}(X^t_{val}, X^t_{tr}, \theta^{t+1}_{s|i}, \theta_j)$ refers to the loss value of task τ_j after the network was updated and trained with the shared parameters for specific task τ_i ($\theta^{t+1}_{s|i}$), and $L_{\tau_j}(X^t_{val}, X^t, \theta^t_s, \theta_j)$ points to the loss value before the update. This intuitively reveals to what extent one task can transfer positive or negative knowledge on the performance of the other task. Note: here the main question of transferability is on the modeling aspect and not the effect of different data representations. This means that different forms of data representation could have various consequences on the multi-task learning results.

Fig. 9 shows the transference from individual tasks to the performance of other tasks over the 30 epochs. The subplots clearly illustrate the constructive impact of training on an individual task on the performance on another task. All sub-plots show that the individual tasks at the beginning have less impact on other tasks, but the impact increases over the training time. Concerning Task 1, initially the task negatively impacts Task 2, while it compensates for this in the later stages so that the result changes to a positive. Focusing on Task 3 transference, we can see Task 2 and Task 4 received a more constructive effect with respect to the other tasks; however, the transference from Task 3 to others reached a similar level in epoch 30. A similar improvement has been transferred from Task 4 to Task 1, Task 3, and Task 5, while the positive impact fluctuated, in particular on Task 2, over the training time.

Eq. (12) was used to quantify the overall transference between the tasks. The box-plots in Fig. 10 show the average impact of different vehicle behaviors (different tasks) on one specific behavior (task). The box-plots show that each task received a positive impact from other tasks. Task 1, Task 3, and Task 4 gained very similar support from other tasks (subplots 10(a), 10(c), and 10(d)). Task 2 obtained more help from Task 1 and Task 3, and less support from Task 4 and Task 5.

These results indicate that some tasks are more constructive than others to the performance on the specific task. This knowledge motivated us to investigate if it is possible to find a sub-optimal group of tasks that should be trained together to increase the performance of the predictive multi-task model. This is described in the next subsection.

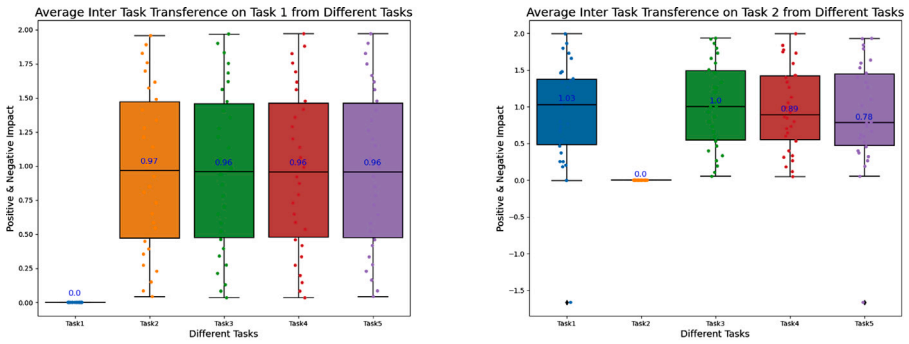
6.2.1. Task grouping results

To answer the question of whether task grouping can positively affect the performance of the predictive model in modeling and predicting the behavior of the vehicles, we built five different groups considering the results obtained in the task transference experiment. In this evaluation, some tasks were appointed to be trained together, and other tasks were used as new features added to the training data. For example, in Group1 (G1), Task 2, Task 3, and Task 4 were considered as the target variable and trained together. In comparison, Task 1 and Task 5 were used as new features in training the data.

Table 7 shows the evaluation result from the five subgroups. Task 4 gained more with respect to the other tasks by 20% (see Table 4), which is a remarkable improvement. Task 1 received 4% and 6% in group2 and group3, respectively. Similar improvements have been obtained for Task 3 in the task grouping. In contrast, the performance did not change for Task 5 in this practice. The numbers received by this experiment suggest that even if the tasks are not highly related, they can positively impact the predictive model's performance if the right combination is found.

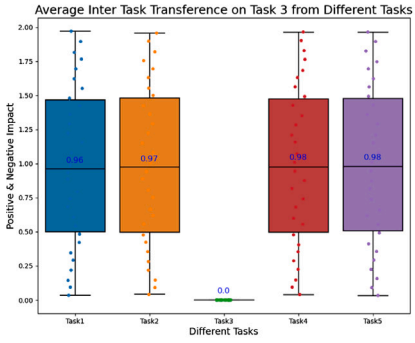
7. Discussion, conclusion, and future work

This study presents a multi-task snapshot-stacked ensemble deep neural network approach for predicting vehicles behavior in the

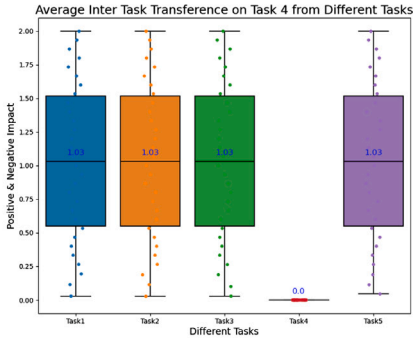


(a) Avg on Task1

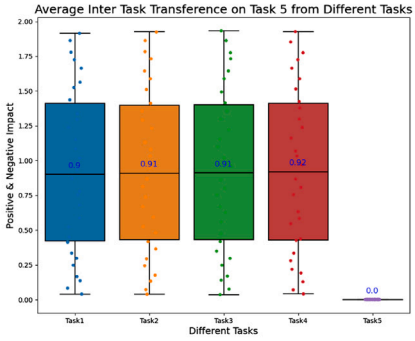
(b) Avg on Task2



(c) Avg on Task3



(d) Avg on Task4



(e) Avg on Task5

Fig. 10. The average task transference/impact over the training process.

automotive sector. The experimental evaluation of thousands of heavy-duty Volvo trucks' logged real data show that the proposed ensemble approach can adequately map the low-resolution data to vehicles' behavior calculated and transferred from high-resolution data. The results indicate that MTSSE can produce a generic multi-task predictive model that performs well on low-resolution data through a single training process for behavior prediction. Regarding the research questions introduced in Section 1, we evaluated the key aspects, which express our approach's efficiency, effectiveness, and generality.

Considering the first objective (RQ1), figures obtained in model performances experiments show a significant difference between the proposed ensemble performance and other ensemble-based approaches. This is observed in both single-task and multi-task formulations. The results described how the meta-learning could learn from the errors received by the snapshots models in the first phase of the ensemble process to predict different vehicles' behavior simultaneously. Indeed, the results in the second phase ensure the stability of the meta-learning performance throughout the training process, taking into account vehicle low-resolution data usage. The generality of the approach was verified on several public datasets. The results demonstrate that MTSSE and STSSE could provide equal and, in most cases, better predictive models than other ensemble-based approaches, both for the multi- and single-task formulation of the approach. Therefore, *the multi-task snapshot-stacked ensemble deep neural network approach can successfully map the low-resolution usage to the vehicle behavior pattern and outperforms the other compared ensemble algorithms. The figures obtained in the last phase also show the potential of building a general multi-task forecasting model for this complex behavioral problem.*

Considering the second objective (RQ2), whether the inter-task transference evaluations explained how the individual tasks influenced

Table 6
The comparison between STSSE and three ensemble approaches on various datasets. We employed 5 × 2 cv paired t-test to examine how significant the difference is between the performance of the two models. It means, every time we compare only two models, one STSSE vs. one ensemble-based classifier (e.g., Bagging). "*" points to the alpha level at 0.1 and "**" refers to the alpha level at 0.05 to reject the null-hypothesis.

#	Dataset	STSSE	Bagging		Boosting		Stacking	
		accuracy	accuracy	t-test * **	accuracy	t-test * **	accuracy	t-test * **
1	VBS fault	**0.96 ± 0.01**	0.93 ± 0.01	2.86 ✓✓	0.94 ± 0.001	−2.47 ✓✓	0.94 ± 0.01	−2.49 ✓✓
2	CIC	**0.76 ± 0.02**	0.73 ± 0.006	1.64 ✓✗	0.73 ± 0.01	1.53 ✓✗	0.73 ± 0.001	1.66 ✓✗
3	Covtype	0.82 ± 0.09	0.68 ± 0.009	2.86 ✓✓	0.86 ± 0.002	−0.87 ✗✗	0.84 ± 0.005	−0.46 ✗✗
4	CMIYC	**0.99 ± 0.01**	0.91 ± 0.01	9.40 ✓✓	0.93 ± 0.001	6.91 ✓✓	0.91± 0.004	9.49 ✓✓
5	KDD	**0.79 ± 0.39**	0.36 ± 0.34	1.63 ✓✗	0.99 ± 0.001	−0.99 ✗✗	0.62 ± 0.43	0.58 ✗✗
6	BPRM	**0.95 ± 0.012**	0.89 ± 0.003	9.01 ✓✓	0.94 ± 0.004	1.09 ✗✗	0.92 ± 0.001	4.86 ✓✓
7	Diabetes	0.72 ± 0.15	0.75 ± 0.03	−0.39 ✗✗	0.72 ± 0.01	0.02 ✗✗	0.75 ± 0.03	−0.37 ✗✗
8	MNIST	0.92 ± 0.01	0.85 ± 0.004	9.43 ✓✓	0.99 ± 0.005	−9.49 ✓✓	**0.99 ± 0.005**	−9.55 ✓✓
9	Musk	**0.99 ± 0.01**	0.99 ± 0.002	−0.6 ✗✗	0.99 ± 0.003	−1.48 ✓✗	0.99 ± 0.02	−0.6 ✗✗
10	Parkinson	**0.96 ± 0.04**	0.73 ± 0.05	6.76 ✓✓	0.90 ± 0.02	2.48 ✓✓	0.77 ± 0.02	7.57 ✓✓
11	Spam	0.85 ± 0.29	0.82 ± 0.21	0.10 ✗✗	0.67 ± 0.10	1.09 ✗✗	**0.98 ± 0.003**	−0.95 ✗✗
12	Waveform	**0.87 ± 0.02**	0.86 ± 0.003	−0.41 ✗✗	0.85 ± 0.003	0.72 ✗✗	0.86 ± 0.01	−0.33 ✗✗
13	Australia	**0.99 ± 0.002**	0.81 ± 0.003	78.6 ✓✓	0.98 ± 0.03	1.25 ✗✗	0.99 ± 0.01	0.50 ✗✗

Table 7
The tasks that are trained together in different experiment groups. The * indicates that the task is included as a new feature in the data for training and not part of the multi-task learning. The ✓indicates the tasks are trained together in multi-task learning.

Group	Tasks					Performance				
	T1	T2	T3	T4	T5	T1	T2	T3	T4	T5
G1	*	✓	✓	✓	*	*	0.99 ± 0.001	0.94 ± 0.02	0.96 ± 0.01	*
G2	✓	*	✓	✓	*	0.93 ± 0.04	*	0.92 ± 0.04	0.95 ± 0.02	*
G3	✓	✓	✓	*	*	0.96 ± 0.02	0.98 ± 0.01	0.95 ± 0.01	*	*
G4	*	*	✓	✓	✓	*	*	0.91 ± 0.03	0.94 ± 0.02	0.87 ± 0.04
G5	*	*	*	✓	✓	*	*	*	0.93 ± 0.04	0.89 ± 0.04

the other tasks' performance over the training process. Assessing the overall impact and the results shown in the box-plots, it is clear that *the tasks have a constructive impact on each other's performance in the multi-task formulation*.

Considering the third objective (R3), the evaluation of task grouping. The experiments show that tasks could contribute better to the performance of the others when they are coupled and trained with the right tasks. Thus, based on the results from the evaluated sub-groups, it is clear that *multi-task formulation can even reach a better performance by finding the optimal combination of tasks to be trained*. The 20% increase in the performance of Task 4 confirmed the potential of optimal task grouping.

The findings of this study also reveal limitations of the proposed approach, which suggest new directions for future research.

The first limitation relates to the challenge of modeling vehicle behavior by taking various contexts into account. In this study, we focused on a time window of one year on whether the vehicle behaved in certain behaviors or not. However, no other context was considered, like location (where vehicles were driven), which highly affects the usage pattern and multi-task predictive models. This strongly encourages us to build up our ensemble approach by incorporating further context knowledge (e.g., location in a short period e.g., one week) to map low-resolution data usage to certain vehicle behaviors. An interesting extension of our work can be on driver profiling. We could utilize the driver class to model vehicles' behaviors in different contexts, leading to identifying good and bad driving styles as an important factor in vehicle performance. Thus, in this matter, driver behavior as extra knowledge input can be used to characterize the style of vehicle usage in different contexts.

The second limitation pertains to the generality assessment. This study utilized the single-task version of the approach (STSSE) to estimate whether the solution is general enough. Further investigation is needed to employ MTSSE on the datasets with multiple target values to evaluate how MTSSE can perform in other contexts.

CRediT authorship contribution statement

Reza Khoshkangini: Conceptualization, Methodology, Implementation, Writing and finalizing the manuscript. **Peyman Mashhadi:** Methodology, Writing and editing. **Daniel Tegnered:** Data curation, Handling. **Jens Lundström:** Project creation, Writing and editing. **Thorsteinn Rögnvaldsson:** Writing and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

Agarap, A. F. (2018). Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.

Alizadeh, M., Rahimi, S., & Ma, J. (2022). A hybrid ARIMA-WNN approach to model vehicle operating behavior and detect unhealthy states. *Expert Systems with Applications*, Article 116515.

Arpit, D., & Bengio, Y. (2019). The benefits of over-parameterization at initialization in deep ReLU networks. arXiv preprint arXiv:1901.03611.

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixao, T. M., & Mutz, F. (2021). Self-driving cars: A survey. *Expert Systems with Applications*, *165*, Article 113816.

Bousonville, T., Dirichs, M., & Krüger, T. (2019). Estimating truck fuel consumption with machine learning using telematics, topology and weather data. In *2019 international conference on industrial engineering and systems management* (pp. 1–6). IEEE.

Cárdenas-Gallo, I., Sarmiento, C. A., Morales, G. A., Bolivar, M. A., & Akhavan-Tabatabaei, R. (2017). An ensemble classifier to predict track geometry degradation. *Reliability Engineering & System Safety*, *161*, 53–60.

Chen, J., Wang, S., He, E., Wang, H., & Wang, L. (2022). Two-dimensional phase lag index image representation of electroencephalography for automated recognition of driver fatigue using convolutional neural network. *Expert Systems with Applications*, *191*, Article 116339.

Choi, E., & Kim, E. (2017). Critical aggressive acceleration values and models for fuel consumption when starting and driving a passenger car running on LPG. *International Journal of Sustainable Transportation*, *11*(6), 395–405.

Chowdhuri, S., Pankaj, T., & Zipser, K. (2019). Multinet: Multi-modal multi-task learning for autonomous driving. In *2019 IEEE winter conference on applications of computer vision* (pp. 1496–1504). IEEE.

Dong, X., Yu, Z., Cao, W., Shi, Y., & Ma, Q. (2020). A survey on ensemble learning. *Frontiers of Computer Science*, *14*(2), 241–258.

Fifty, C., Amid, E., Zhao, Z., Yu, T., Anil, R., & Finn, C. (2021). Efficiently identifying task groupings for multi-task learning. *Advances in Neural Information Processing Systems*, *34*.

Fu, H., Li, C., Liu, X., Gao, J., Celikyilmaz, A., & Carin, L. (2019). Cyclical annealing schedule: A simple approach to mitigating kl vanishing. arXiv preprint arXiv:1903.10145.

Gao, J., Chen, H., Liu, Y., Li, Y., Li, T., Tu, R., Liang, B., & Ma, C. (2021). The effect of after-treatment techniques on the correlations between driving behaviours and NOx emissions of passenger cars. *Journal of Cleaner Production*, *288*, Article 125647.

González, S., García, S., Del Ser, J., Rokach, L., & Herrera, F. (2020). A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Information Fusion*, *64*, 205–237.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026–1034).

Heidemann, L., Schwaiger, A., & Roscher, K. (2021). Measuring ensemble diversity and its effects on model robustness.

Hou, Y., Edara, P., & Sun, C. (2015). Situation assessment and decision making for lane change assistance using ensemble learning methods. *Expert Systems with Applications*, *42*(8), 3875–3882.

Huang, G., Li, Y., Pleiss, G., Liu, Z., Hopcroft, J. E., & Weinberger, K. Q. (2017). Snapshot ensembles: Train 1, get m for free. arXiv preprint arXiv:1704.00109.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456). PMLR.

Kumar, S. K. (2017). On weight initialization in deep neural networks. arXiv preprint arXiv:1704.08863.

Lattanzi, E., & Freschi, V. (2021). Machine learning techniques to identify unsafe driving behavior by means of in-vehicle sensor data. *Expert Systems with Applications*, *176*, Article 114818.

Le, V., Yao, X., Miller, C., & Tsao, B.-H. (2020). Series DC arc fault detection based on ensemble machine learning. *IEEE Transactions on Power Electronics*, *35*(8), 7826–7839.

Li, Z., Gong, J., Lu, C., & Yi, Y. (2021). Interactive behavior prediction for heterogeneous traffic participants in the Urban road: A graph-neural-network-based multitask learning framework. *IEEE/ASME Transactions on Mechatronics*, *26*(3), 1339–1349.

Li, Z., Wu, D., Hu, C., & Terpenny, J. (2019). An ensemble learning-based prognostic approach with degradation-dependent weights for remaining useful life prediction. *Reliability Engineering & System Safety*, *184*, 110–122.

Lin, N., Zong, C., Tomizuka, M., Song, P., Zhang, Z., & Li, G. (2014). An overview on study of identification of driver behavior characteristics for automotive control. *Mathematical Problems in Engineering*, *2014*.

Liu, P., Kurt, A., & Özgüner, Ü. (2014). Trajectory prediction of a lane changing vehicle based on driver behavior estimation and classification. In *17th international IEEE conference on intelligent transportation systems* (pp. 942–947). IEEE.

Marina Martinez, C., Heucke, M., Wang, F.-Y., Gao, B., & Cao, D. (2018). Driving style recognition for intelligent vehicle control and advanced driver assistance: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *19*(3), 666–676. http://dx.doi.org/10.1109/TITS.2017.2706978.

Miyajima, C., & Takeda, K. (2016). Driver-behavior modeling using on-road driving data: A new application for behavior signal processing. *IEEE Signal Processing Magazine*, *33*(6), 14–21. http://dx.doi.org/10.1109/MSP.2016.2602377.

Mondal, S., & Gupta, A. (2022). Evaluation of driver acceleration/deceleration behavior at signalized intersections using vehicle trajectory data. *Transportation Letters*, 1–13.

Mosavi, A., Sajedi Hosseini, F., Choubin, B., Goodarzi, M., Dineva, A. A., & Rafiei Sardooi, E. (2021). Ensemble boosting and bagging based machine learning models for groundwater potential prediction. *Water Resources Management*, *35*(1), 23–37.

Mousavi, M., Moradi, M., Chaibakhsh, A., Kordestani, M., & Saif, M. (2020). Ensemble-based fault detection and isolation of an industrial gas turbine. In *2020 IEEE international conference on systems, man, and cybernetics* (pp. 2351–2358). IEEE.

Pentland, A., & Liu, A. (1999). Modeling and prediction of human behavior. *Neural Computation*, *11*(1), 229–242.

Powell, S., Cezar, G. V., & Rajagopal, R. (2022). Scalable probabilistic estimates of electric vehicle charging given observed driver behavior. *Applied Energy*, *309*, Article 118382.

Prakash, S., & Bodisco, T. A. (2019). An investigation into the effect of road gradient and driving style on NOX emissions from a diesel vehicle driven on urban roads. *Transportation Research Part D: Transport and Environment*, *72*, 220–231.

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

Shahverdy, M., Fathy, M., Berangi, R., & Sabokrou, M. (2020). Driver behavior detection and classification using deep convolutional neural networks. *Expert Systems with Applications*, *149*, Article 113240.

Wang, Z., Liao, X., Wang, C., Oswald, D., Wu, G., Boriboonsomsin, K., Barth, M. J., Han, K., Kim, B., & Tiwari, P. (2020). Driver behavior modeling using game engine and real vehicle: A learning-based approach. *IEEE Transactions on Intelligent Vehicles*, *5*(4), 738–749. http://dx.doi.org/10.1109/TIV.2020.2991948.

Xie, J., Hu, K., Li, G., & Guo, Y. (2021). CNN-based driving maneuver classification using multi-sliding window fusion. *Expert Systems with Applications*, *169*, Article 114442. http://dx.doi.org/10.1016/j.eswa.2020.114442, URL https://www.sciencedirect.com/science/article/pii/S0957417420311003.

Xing, Y., Lv, C., Cao, D., & Velenis, E. (2020). A unified multi-scale and multi-task learning framework for driver behaviors reasoning. arXiv preprint arXiv:2003.08026.

Xing, Y., Lv, C., Wang, H., Cao, D., & Velenis, E. (2020). An ensemble deep learning approach for driver lane change intention inference. *Transportation Research Part C (Emerging Technologies)*, *115*, Article 102615.

Xu, Z., Wei, T., Easa, S., Zhao, X., & Qu, X. (2018). Modeling relationship between truck fuel consumption and driving behavior using data from internet of vehicles. *Computer-Aided Civil and Infrastructure Engineering*, *33*(3), 209–219. http://dx.doi.org/10.1111/mice.12344, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/mice.12344.

Xu, Y., Zheng, Y., & Yang, Y. (2021). On the movement simulations of electric vehicles: A behavioral model-based approach. *Applied Energy*, *283*, Article 116356. http://dx.doi.org/10.1016/j.apenergy.2020.116356, URL https://www.sciencedirect.com/science/article/pii/S0306261920317360.

Xun, Y., Liu, J., & Shi, Z. (2020). Multitask learning assisted driver identity authentication and driving behavior evaluation. *IEEE Transactions on Industrial Informatics*, *17*(10), 7093–7102.

Yang, S., Shao, Q., & Bian, C. (2022). Reliability analysis of ensemble fault tolerance for soft error mitigation against complex radiation effect. *Reliability Engineering & System Safety*, *217*, Article 108092.

Yao, W., Zhao, H., Davoine, F., & Zha, H. (2012). Learning lane change trajectories from on-road driving data. In *2012 IEEE intelligent vehicles symposium* (pp. 885–890). IEEE.

Zhang, H., & Fu, R. (2021). An ensemble learning-online semi-supervised approach for vehicle behavior recognition. *IEEE Transactions on Intelligent Transportation Systems*, 1–17. http://dx.doi.org/10.1109/TITS.2021.3095053.

Zhang, D., & Gao, Z. (2021). An ensemble approach for fault diagnosis via continuous learning. In *2021 IEEE 19th international conference on industrial informatics* (pp. 1–5). IEEE.

Zhang, Y., & Yang, Q. (2021). A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*.

Zhou, Z.-H. (2021). Ensemble learning. In *Machine learning* (pp. 181–210). Springer.